CYCLIC CONVOLUTION OF REAL SEQUENCES : HARTLEY VERSUS FOURIER AND NEW SCHEMES

P. DUHAMEL* and M. VETTERLI**

* CNET/PAB/RPE 38-40, rue du Général Leclerc, F-92131 Issy-les-Moulineaux (France)

** Ecole Polytechnique Fédérale de Lausanne 16, Chemin de Bellerive, CH-1007 Lausanne (Suisse)

ABSTRACT

Recently, new fast transforms (such as the discrete Hartley transform in particular) have been proposed which are best suited for the computation of cyclic convolution of real sequences. Two approaches using Fourier or Hartley transforms are first compared, showing that the recently proposed FFT algorithms for real data present a lower arithmetic complexity than the corresponding DHT-based approach. Improvements are made to both types of algorithms, leading to different trade offs between arithmetic and structural complexity. We also present a new Hartley Transform algorithm with lower arithmetic complexity than any previously published one.

I. INTRODUCTION

The radix-2 Fast Fourier Transform algorithm was first explained by Cooley and Tukey in 1965, in a version on complex data [1]. But, as most of the data to be treated is real, instead of complex, the need of a version of the FFT on real data removing extra calculations from the complex case became soon apparent, and such a program was quickly published by Bergland [3].

In fact this program does not seem to have been widely used, and many people preferred either to compute a length 2^n- real DFT either by the use of a length 2^{n-1} complex FFT plus some additional operations, or to compute two real FFT's at a time by using one length 2^n complex FFT.

More recently,the 2^n FFT algorithms with the minimum known number of both multiplications and additions were also proposed in a real-data version [6,7,10,11].

In the same time, Bracewell [8] proposed the use of fast Discrete Hartley Transforms (FDHT), which is real of nature, as a substitute to the FFT for cyclic convolutions on real data.

This paper is concerned with the computation of cyclic convolutions of length -2^n real data through the use of fast transforms.

The well-known approach of computing real convolutions through FFT algorithms on complex data will not be considered here, and we shall only consider programs specially designed for real data, since they allow to obtain the lowest known number of arithmetic operations, without increase in programlength.

In the first paragraph, we shall first briefly present the two basic approaches considered here : convolution using FFT's, and convolution using FDHT's. This comparison shows that the former has some advantages when considering arithmetic complexity,while the latter has a simpler structure, since the DHT is self-inverse.

The second paragraph will present a scheme for cyclic convolution using two forward DFT's on real data, and a small number of additional operations. This solution has the advantage of a simple structure since it uses only forward DFT's on real data.

The third paragraph will present a hybrid FFT/FDHT algorithm, allowing to obtain any length -2^n FDHT using only 2 additions more than the best FFT algorithms known for real data. In this paragraph, we also show how to derive very quickly improved FDHT algorithms from their FFT counterparts.

II. THE INITIAL SCHEMES

II.1. Convolution using FFT's

Since the DFT has the convolution property, the convolution scheme has the structure given in fig. 1. When the initial sequences $\{x_n\}$ and $\{h_n\}$ are real, $\{X_h\}$, $\{H_h\}$, and $\{Y_h\}$ have hermitian symmetry : $Y_h = Y^*_{-k}$, and the forward DFT has to be performed on real data, while the inverse FFT has to be performed on data with hermitian symmetry. Both kind of FFT's have been published [7,11] have the same arithmetic complexity, and can be performed in place. (In fact, an FFT on data with hermitian symmetry can be derived by reversing the flow-graph of an FFT on real data).

If the FFT's are performed with the lowest arithmetic complexity, using FFCT [6] or split-radix [7] algorithms, this leads to the following number of operations :

$$(1) \quad M_n^{\text{conv. FFT}} = 2\,M_n^{fr} + 2 + 3\,.\,(2^{n-1} - 1)$$

$$= 2^{n-1}(2n-3) + 3$$

$$(2) \quad A_n^{\text{conv. FFT}} = 2\,A_n^{fr} + 3\,(2^{n-1} - 1)$$

$$= 2^{n-1}\,(6n - 7) + 5$$

In these operation counts, it has been taken into account that, due to the symmetry of both $\{X_k^f\}$ and $\{H_k^f\}$, $\{Y_k^f\}$ can be obtained with 2 real multiplications for k=0, and k=N/2, and N/2-1 complex multiplications (3 real mults + 3 real adds each) for k=1,---N/2-1.

CH2243-4/86/0000-0229 $1.00

This arithmetic complexity is rather low, as shown in table 1, since it is based on powerful FFT algorithm on real data, but a disadvantage of this scheme is that it needs both forward transform on real data and inverse transform on complex data with hermitian symmetry. This results in an increase in program length for the cyclic convolution.

II.2. Convolution using FDHT's

The Discrete Hartley Transform is defined as the sum of the real and imaginary part of the Fourier Transform :

$$(3) \quad X_k^h = \sum_{n=0}^{N-1} x_n(\cos\frac{2\pi nk}{N} + \sin\frac{2\pi nk}{N})$$

As can be seen from the definition, the DHT is its own inverse, and is a real transform. Furthermore, the convolution in the time domain corresponds to the following operation in the Hartley domain :

$$(4) \quad Y_k^h = X_k^h \cdot H_e^h(k) + X_{-k}^h H_o^h(k)$$

where $H_e^h(k)$ (resp. $H_o^h(k)$) is the even (resp. odd) part of the Hartley transform of $\{h_n\}$.

So, the convolution scheme using Hartley transforms becomes as shown in fig. (2).

When considering eq.(4), the "multiplications" in the Hartley domain seems to need 2 multiplications per point, but it is easy to see that, by grouping the computation of Y_k^h and Y_{-k}^h, this number can be reduced to 3/2 multiplications per point :

$$(5) \quad \begin{bmatrix} Y_k^h \\ Y_{-k}^h \end{bmatrix} = \begin{bmatrix} H_e(k) & H_o(k) \\ -H_o(k) & H_e(k) \end{bmatrix} \begin{bmatrix} X_k^h \\ X_{-k}^h \end{bmatrix}$$

and eq.(5) is easily recognized as a complex multiplication, needing only 3 real multiplications and 3 additions.

If we consider also that :

$$(6) \quad \begin{aligned} Y_o^h &= X_o^h \; H_e(o) \\ Y_{N/2}^h &= X_{N/2}^h \; H_e(N/2) \end{aligned}$$

One can see that the set of "multiplications" in the Hartley domain needs exactly the same number of operations as in the Fourier domain.

The remaining part of the scheme given in fig. (2) to be evaluated consists in the FDHT.

FDHT was first introduced by Bracewell [8] in a radix 2, decimation in time version, and Sorensen, Jones, Burrus and Heideman [9] recently proposed different types of FDHT algorithms.

Inspection of these algorithms allows the following conclusions :

A radix 2 FDHT algorithm needs N-2 additions more than the corresponding FFT algorithm on real data, and the same number of multiplications.

The split-radix algorithms, which seems already to be one of the best compromises for real-data FFT's (lowest known number of operations, in-place computation, compact program) gives also the lowest known number of operations for the FDHT, but requires $2(\frac{2^{n-1}-(-1)^{n-1}}{3})$ additions more than the corresponding split-radix FFT algorithm on real data.

If the FDHT in fig.(2) is computed through a split-radix algorithm, this gives the following arithmetic complexity for the cyclic convolution :

$$(7) \quad M_n^{\text{conv. DHT}} = 2^{n-1}(2n-3) + 3$$

$$(8) \quad A_n^{\text{conv. DHT}} = 2^{n-1}(6n-7) + 5 + 4\,\frac{2^{n-1}-(-1)^{n-1}}{3}$$
$$= 2^{n-1}(6n-\tfrac{17}{3}) + 5 - \tfrac{4}{3}(-1)^{n-1}$$

One can see that convolution using FDHT's will have a simpler structure, even if the FDHT programs have the same code length as FFT programs, due to the self-inverse property of the Hartley transform. This simplicity will be obtained at the cost of a small increase of the number of additions.

In the following, we shall improve both solutions :

III. A MORE REGULAR CONVOLUTION SCHEME USING FFT's

The improvement allowing to use two forward FFT's in the convolution scheme is based on some of the remarks made when studying FDHT's :

The convolution property in the Hartley domain, as given in eq.(4) can be rewritten as follows, by reversing the roles of X_k and H_k :

$$(9) \quad Y_k^h = X_e^h(k) \cdot H_k^h + X_o^h(k)\, H_{-k}^h$$

But, $X_e^h(k)$, being the even part of the Hartley transform of $\{x_n\}$ is also the real part of the DFT of $\{x_n\}$, and $X_o^h(k)$ is the corresponding imaginary part. We can then rewrite eq. (5) as follows :

$$(10) \quad \begin{bmatrix} Y_k^h \\ Y_{-k}^h \end{bmatrix} = \begin{bmatrix} H_k^h & H_{-k}^h \\ H_{-k}^h & -H_k^h \end{bmatrix} \begin{bmatrix} \mathrm{Re}X_k^f \\ \mathrm{Im}X_k^f \end{bmatrix}$$

which is also equivalent to a complex product.

Furthermore, since a DHT is the sum of the real and imaginary part of a DFT, the last FDHT in fig.(2) can be replaced by an FFT followed by the sum of the real and imaginary part of each DFT component, thus resulting in the diagram of fig. (3).

This scheme requires exactly the same number of operations as the usual convolution scheme based on FFT's, plus N-2 additions due to the +/- box :

$$(11) \quad A_n^{\text{conv. fft}} = 2^{n-1}(6n-5) + 3$$

This structure is now as regular as the one using FDHT's, to the price of about $2^n/3$ extra additions compared to the FDHT-scheme. It has the advantage of using only usual FFT algorithms on real data, that should become widely used, due to their simplicity and compactness.

IV. IMPROVED FDHT ALGORITHM

FDHT algorithms, as found in [8], and [9] can be improved by making full use of the close relationship between Fourier and Hartley transforms : In fact, nearly all the extra additions needed to compute FDHT's instead of FFT's can be nested inside the twiddle factors of the FFT, thus resulting in hybrid FDHT/FFT algorithm.

$$(12)\quad X_k^f = \sum_{n=0}^{N-1} x_n(\cos\frac{2\pi nk}{N} - j\sin\frac{2\pi nk}{N}) = \sum_{h=0}^{N-1} x_n W_N^{nk}$$

By multiplying the DFT of $\{x_n\}$ by (1+j), we obtain eq. (13) :

$$(13)\quad (1+j)\,X_k^f = (\mathrm{Re}\,X_k^f - \mathrm{Im}\,X_k^f) + j\,(\mathrm{Re}\,X_k^f + \mathrm{Im}\,X_k^f)$$
$$= X_k^h + j\,X_{-k}^h$$

which means that multiplication of X_k^f by (1+j) gives us the Hartley transform of $\{x_n\}$.

Let us apply this remark to the basic split-radix decimation-in-time decomposition of the DFT [10] as given in (14)

$$(14)\quad X_k^f = \sum_{k=0}^{N/2-1} x_{2n} W_N^{2nk} + W_N^k \sum_{n=0}^{N/4-1} x_{4n+1} W_N^{4nk} +$$
$$= W_N^{3k} \sum_{n=0}^{N/4-1} x_{4n+3} W_N^{4nk}$$

$$(15)\quad (1+j)\cdot X_k^f = X_k^h + j\,X_{-k}^h = (1+j)\sum_{n=0}^{N/2-1} x_{2n} W_N^{2nk}$$
$$+ (1+j)\,W_N^k \sum_{n=0}^{N/4-1} x_{4n+1} W_N^{4nk} +$$
$$+ (1+j)\,W_N^{3k} \sum_{n=0}^{N/4-1} x_{4n+3} W_N^{4nk}$$

eq. (15) is now the basic recursion of the improved FDHT algorithm : it transforms the length-N DHT into a length N/2. DHT plus two length N/4 - DFT's, and some multiplications by twiddle factors of the form $(1+j)\,W_N^k$ or $(1+j)\,W_N^{3k}$.

As in the case of FFT's on real data, the butterflies with twiddle factors W_N^k have to be computed together with the ones with twiddle factors W_N^{-k}. This needs (N/8 - 1) general 6-mult-18 adds butterflies, plus one 0-mult-6-adds butterfly (corresponding to k=0) and one 2-mult-4-adds butterfly (corresponding to k = N/8).

Iteration of these addition counts gives exactly the same number than was obtained for FFT's on real data, except for the FDHT of length 4 which needs 2 extra addition.

This leads to :

$$(16)\quad A_n^{hart} = 2^{n-1}(3.2^n - 5) + 6$$

and the whole convolution with this algorithm will need :

$$(17)\quad A_n^{conv\ hart.} = 2^{n-1}(6_n - 7) + 9$$

The number of multiplications is the same as for the other schemes.

The convolution scheme using this new fast Hartley transform algorithm has now the same number of multiplications, and 4 additions more than the best known scheme, with the advantage of using only a single DHT program, to be used twice.

It must be noticed that this has to be paid by an increase in the program length. The new FDHT algorithm has the same structure as the corresponding FFT algorithm, some of the butterflies in the FFT program being replaced by the ones corresponding to eq.(15). Nevertheless, both programs will be the same size when using autogen techniques.

Another remark is that the same type of improvement, if applied to other classical decompositions (eg. radix 2, 4, etc...), improves all existing programs.

V. CONCLUSION

In this paper, we have first presented two usual schemes for cyclic convolution of real signals, via FFT or FDHT. Operation counts are given for both of them using the fastest known algorithms, showing that the FFT-based convolution has a lower arithmetic complexity, but a more complex structure.

Improvements are then made to both schemes, since we propose :

- a FFT-based scheme using two forward transform on real data.
- a FDHT-based algorithm with reduced number of additions.

All operation counts are summarized in Table 1 allowing to choose the best tradeoff between low arithmetic complexity and structural complexity.

REFERENCES

[1] J.W. COOLEY, J.W. TUKEY : "An algorithm for machine computation of complex Fourier series". Mat Comput. 1965, vol 19, pp. 297-301.

[2] R.D. PREUSS : "Very fast computation of the radix 2 Discrete Fourier Transform". IEEE Trans. on ASSP, 1982, vol. 30, pp. 595-607.

[3] G.D. BERGLAND "A Fast Fourier Transform Algorithm for real-valued series". Commun. ACM, vol. 11, pp. 703-710, Oct. 1968.

[4] H. ZIEGLER. "A fast Fourier Transform Algorithm for symmetric real-valued series". IEEE trans. on Audio and Electroacoustics, vol. AU-20, n°5, pp. 353-356, Dec. 1972.

[5] J.B. MARTENS : "Discrete Fourier Transform Algorithms for real-valued sequences". IEEE Trans. on ASSP, vol. ASSP-32, n°2, pp. 390-396, April 1984.

[6] M. VETTERLI, H.J. NUSSBAUHER : "Simple FFT and DCT algorithms with reduced number of operations". Signal Processing, vol. 6, n°4, pp. 267-278, July 1984.

[7] P. DUHAMEL, H. HOLLMANN "Implementation of "split-radix" FFT algorithms for complex, real, and real-symmetric data". ICASSP'85, Tampa.

[8] R.N. BRACEWELL "The Fast Hartley Transform" Proc IEEE. Vol 22, n°8, Aug. 1984, pp. 1010-1018.

[9] H.V. SORENSEN, D.L. JONES, C.S. BURRUS, M.T. HEIDEMAN. "On computing the Discrete Hartley Transform". IEEE Trans. ASSP, vol. ASSP-33, n°4, pp. 1231-1238, Oct. 1985.

[10] P. DUHAMEL "Un algorithme de transformation de Fourier rapide à double base". Ann. Télécom. Vol. 40, n°9-10, Sept-Oct. 1985, pp. 481-494.

[11] M. VETTERLI, H.J. NUSSBAUMER "Algorithmes de transformation de Fourier et en cosinus mono et bi-dimensionnels"". Ann. Télécom. Vol. 40, n°9-10, pp. 466-476, Sept-Oct 1985.

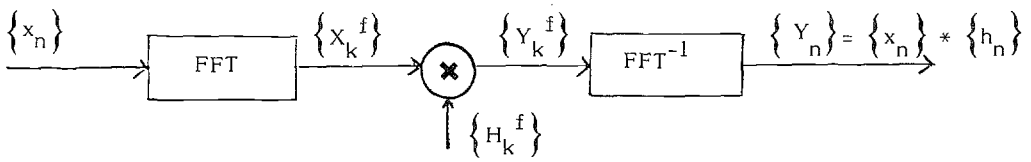

Fig. 1 : Usual convolution scheme using FFT's.

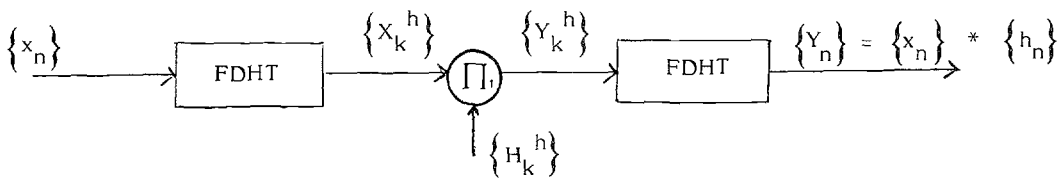

Fig. 2 : Convolution scheme based on FDHT's.

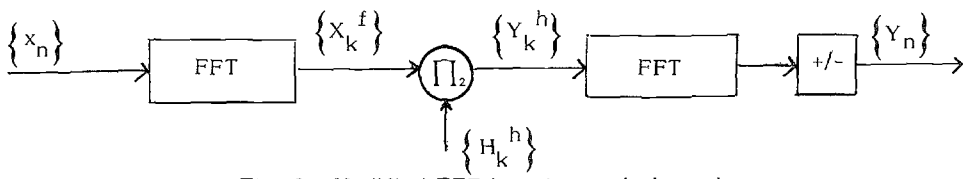

Fig. 3 : Modified FFT-based convolution scheme using only forward FFT's on real data.

N	number of mult.	Number of additions: initial FFT - based	initial FDHT - based	FFT - based with 2 forward FFT's	improved FDHT - based
8	15	49	53	55	53
16	43	141	153	155	145
32	115	373	393	403	377
64	291	933	977	995	937
128	707	2245	2329	2371	2249
256	1667	5253	5425	5507	5257
512	3843	12037	12377	12547	12041
1024	8707	27141	27825	28163	27145
2048	19459	60421	61785	62467	60425

Table 1 : Number of arithmetic operations for a cyclic convolution on real data.